12. Стандартен конзолен вход и изход в C++

Проф. д-р Емил Хаджиколев

1. Конзола;
2. Потоци;
3. Библиотеки;
4. Оператори.

Задаване на входни данни за програмата

- **Програмата може да получава входни данни** преди стартиране или по време на работа (в run-time), без да е необходимо да се компилира наново. Това може да стане по различни начини
  - **задаване на параметри при стартиране**;
  - **въвеждане от потребителя**
    - през **конзолата** или
    - чрез **графичен интерфейс**
  - четене от **файл, база данни, уеб** (или друга) **услуга** и др.

Конзола

- Системна конзола или само конзола е физическо устройство състоящо се от клавиатура и екран.
- Предоставя възможност за комуникация с операционната система (ОС) чрез
  - въвеждане на текст от клавиатурата и
  - извеждане на входна и изходна информация на екрана.
- Конзолата е първият създаден начин за комуникация с ОС чрез Command Line Interface (CLI; сега ОС имат и графичен интерфейс) ;
- В конзолата се задават команди към ОС;
- Има множество стандартни команди – изпълними програми.
- Създадените от нас изпълними програми (.exe) също стават част от командите, които може да извикваме в командния ред.

Достъп до конзолата

- За достъп до конзолата се използват различни обвивки (shell; команден интерпретатор; също изпълними програми), които интерпретират текста въведен от командния ред (command line interpreter).
- Често самата обвивка се нарича конзола, но тя е само средство за достъп до нея;
- В обвивките може да се ползват стандартни команди и конструкции за управление, които могат да бъдат комбинирани в скриптове;
- Примерни обвивки
  - DOS Prompt – когато нямаше Windows
  - Command Prompt в Windows (cmd.exe) – емулира действието на DOS Prompt.
  - PowerShell в по-новите версии на Windows – предоставя повече стандартни команди и конструкции;
  - Bash shell в Linux.

Стартиране на програми в командния интерпретатор

- Може да стартираме създадените от нас изпълними програми в командния интерпретатор (без да средствата – напр., Visual Studio – с които сме създали програмите):
  - Стартираме cmd.exe
  - В командния ред записваме относително или абсолютно име на нашата програма
  - След това може да запишем параметри (разделени с интервали), които да използваме в нашата програма:

  <програма> [параметър1] [параметър2][… параметърN]

Четене на параметри от командния ред в C++

- За да използваме параметри от командния ред трябва да зададем специален main-метод – с два формални параметъра (които получават стойност автоматично при стартиране на програмата с параметри):

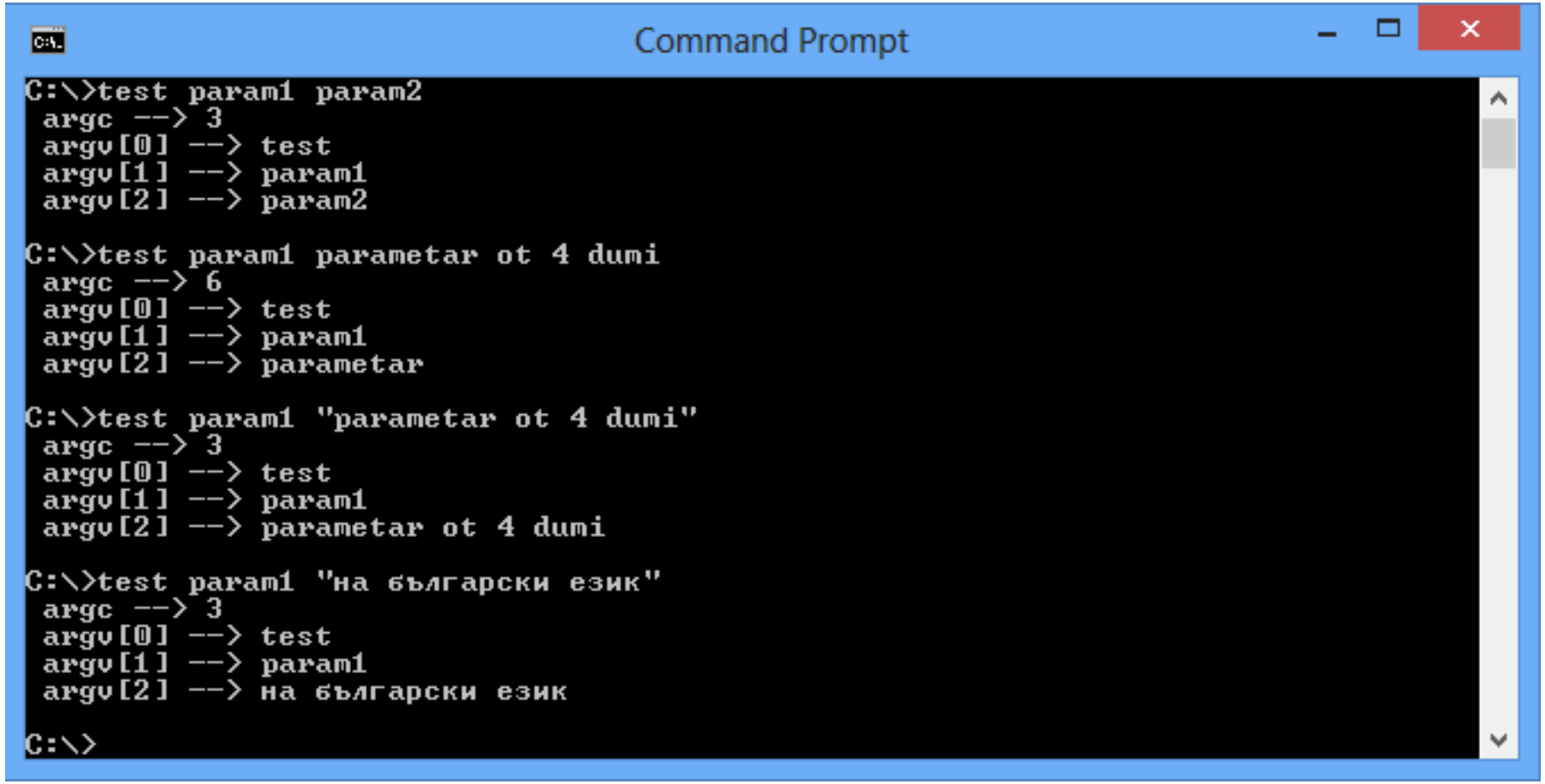
```
int main(int argc, char* argv[])
```

  - В първия параметър се получава броя на параметрите
  - Вторият параметър е масив от низове (от тип char)
  - Всеки елемент на масива съдържа параметъра като низ.
    - argv[0] е първият аргумент – който всъщност е името на програмата;
    - argv[1] е вторият аргумент;
    - argv[argc-1] е последният аргумент;
    - ако достъпим несъществуващ елемент се получава грешка при изпълнение на програмата.

Четене на параметри от командния ред в C++ - пример

```
#include <iostream>
#include <string>
using namespace std;


int main(int argc, char* argv[])
{
	setlocale(LC_ALL, "bg");

	cout << " argc --> " << argc << '\n';		// брой аргументи/параметри зададени от командния ред

	string first_arg = argv[0];			// първи аргумент

	cout << " argv[0] --> " << first_arg << '\n'; // първи аргумент
	cout << " argv[1] --> " << argv[1] << '\n';   // втори аргумент
	cout << " argv[2] --> " << argv[2] << '\n';   // трети аргумент

	// при достъп до несъществуващ елемент - грешка при изпълнение

	return 0;
}
```

Четене на параметри от командния ред в C++ - примерни извиквания на програмата test.exe



Комуникация с конзолата

- Комуникацията с конзолата се извършва чрез абстракции наречени потоци:
  - поток за изход;
  - поток за вход;
  - поток за грешки.
- В езиците за програмиране са създадени механизми (обекти или други начини) за потоците, с помощта на които въвеждаме и извеждаме данни в конзолата.
- В C++ обектите за входно-изходни потоци са:
  - cout – изход;
  - cin – вход;
  - cerr – грешки.

Оператори за вход и изход към конзолата

- Операторите за вход и изход се използват съвместно с обектите за потоци, дефинирани в стандартния namespace std;
- Оператор за изход - << - пренасочва данните от програмата към указан изходен поток:

```
cerr << " тест на потока за грешка" << '\n';
cout << " Въведете число: ";
```

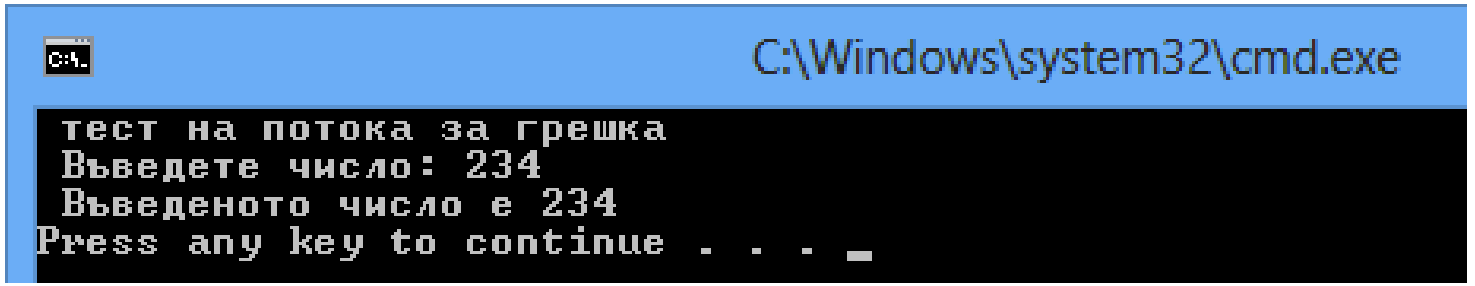
- Оператор за вход - >> - пренасочва входния поток към променливи на програмата:

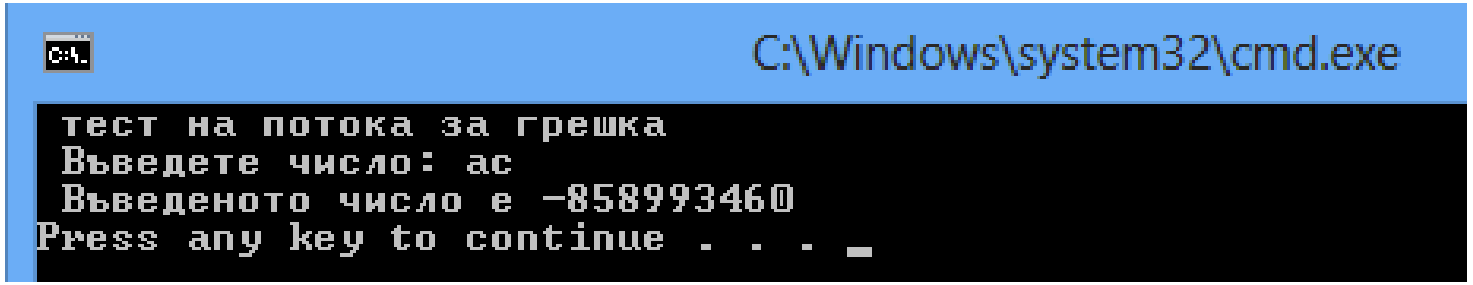
```
cin >> i
```

Пример за вход и изход

```
#include <iostream>
#include <string>
using namespace std;
int main(){
	setlocale(LC_ALL, "bg");

	cerr << " тест на потока за грешка" << '\n';
	cout << " Въведете число: "; // без край на ред - за да остане маркера на същия ред

	int i;

	cin >> i; // Програмата спира и изчаква въвеждане на данни от клавиатурата
		  // След натискане на Enter потока се обработва като се задава стойност на i
		  // надяваме се да е коректна

	cout << " Въведеното число е " << i << '\n';
	return 0;
}
```

Примерна работа с програмата





- Стандартните потоци за вход, изход и грешки са насочени към конзолния терминал, но могат да се пренасочват към файлове и др.

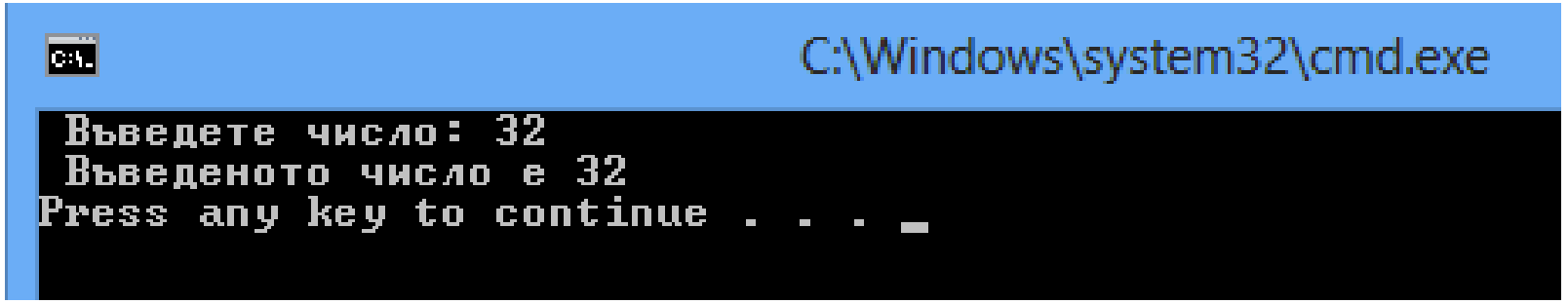
Преобразуване на низ до число

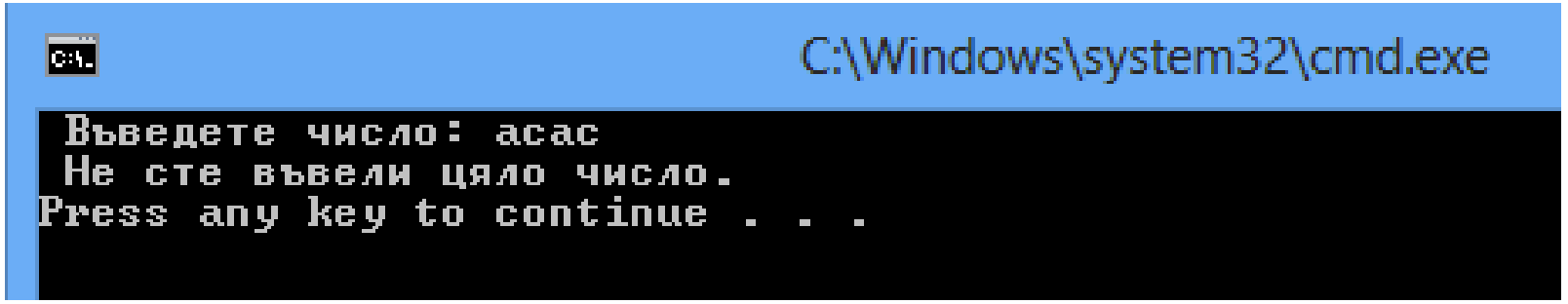
- Съществуват специални функции за преобразуване на низ от тип string (зададен като параметър, въведен от конзолата или по друг начин) до примитивен тип:
  - stoi(низ) – връща число от тип int;
  - stof(низ) – връща число от тип float;
  - stod(низ) – връща число от тип double;
  - stol(низ) – връща число от тип long;
  - stoll(низ) – връща число от тип long long.
- Функциите генерират изключение, ако параметъра-низ не може да се преобразува до съответния тип.
- За изключения няма да говорим сега – само ще покажем как се обработват – в try-catch блокове;

Преобразуване на низ до число - пример

```
#include <iostream>
#include <string>
using namespace std;
int main(){
	setlocale(LC_ALL, "bg");

	cout << " Въведете число: ";
	string s;
	cin >> s; //  Въвеждане от конзолата

	try {
		int i = stoi(s);  // преобразува низ s до цяло число, което връща като резултат
		cout << " Въведеното число е " << i << '\n';
	}
	catch (invalid_argument e){
		cerr << " Не сте въвели цяло число. " << '\n';
	}
	return 0;
}
```

Примерна работа с програма

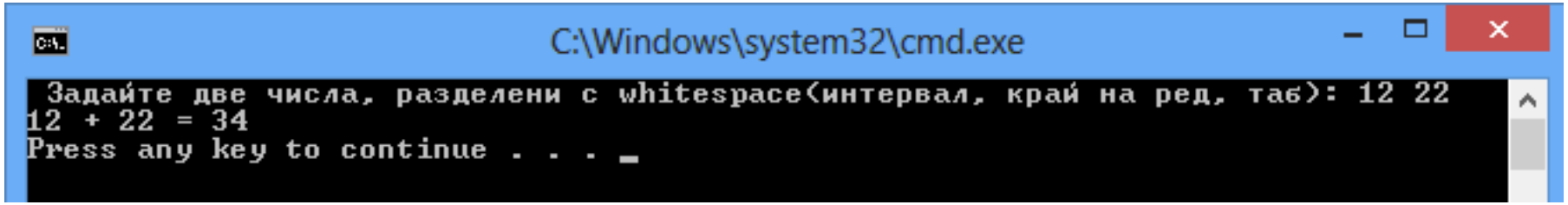




Едновременно въвеждане на няколко променливи от входния поток

- Подобно на едновременното извеждане в потока cout, може едновременно да се въведат няколко променливи чрез cin:

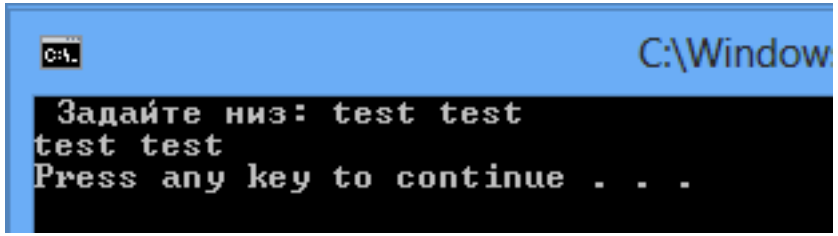
```
cout << " Задайте две числа, разделени с
whitespace(интервал, край на ред, таб): ";

int i, j;
cin >> i >> j; //  Въвеждане от конзолата
cout << i << " + " << j << " = " << i+j << '\n';
```



Четене на низ с интервали

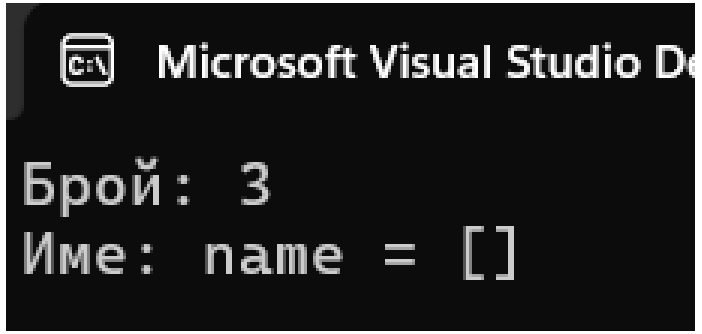
- Със стандартния оператор за четене от конзолата в една променлива се прочита дума (без интервали и други бели символи).
- С функция getline(cin, низ) може да прочетем въведен низ заедно с интервалите.

```
string s;
getline(cin, s);
cout << s << '\n';
```



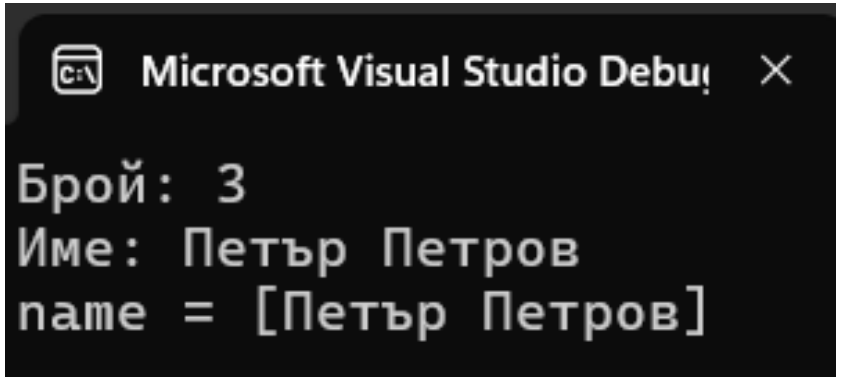
Грешки при „getline след >> “

```cpp
#include <iostream>
#include <windows.h> // за български – Visual Studio
#include <string>
using namespace std;

int main() {
    SetConsoleOutputCP(1251);
    SetConsoleCP(1251);

    int n;
    string name;

    cout << "Брой: ";
    cin >> n;                  // оставя '\n' в буфера

    cout << "Име: ";
    getline(cin, name);        // ЧЕТЕ празния ред -> name == ""
    cout << "name = [" << name << "]\n";
}
```



Решение 1 – „getline след >> “

```cpp
#include <iostream>
#include <windows.h> // за български – Visual Studio
#include <string>

using namespace std;

int main() {
    SetConsoleOutputCP(1251);
    SetConsoleCP(1251);

    int n;
    string name;

    cout << "Брой: ";
    cin >> n;                  // оставя '\n' в буфера

    cout << "Име: ";
    getline(cin >> ws, name); // ws премахва '\n' и всички водещи интервали
    cout << "name = [" << name << "]\n";
}
```



Решение 2 – „getline след >> “

- Не се ползва ”cin >>”.
- Използва се само getline.
- Прочетения текст се парсира/преобразува до желания тип – напр. число.